**CSCA20 Lab 2 (Week 3)**

**Objectives**

1. Practise working with variables
2. Practise defining functions in the editor
3. Practice writing boolean expressions and conditional statements
4. Use the stylechecker

**Variables**

The values in the expressions that we are working with often have more meaning than the numbers alone reveal. For example, the expression `0.5 * 10 * 5`, might be the area of a triangle, where the value 10 is the base and 5 is the height. To associate a name with those values, we use variables.

The general form of an assignment statement is: `variable = expression`

In the shell:

- Assign the value 10 to a variable named `base` and the value 5 to a variable `height`
- Now rewrite the expression `0.5 * 10 * 5` so that it uses variables `base` and `height`
- Next, assign the expression to a variable named `area`
- Reassign variables `base` and `height` the values 12 and 17, respectively. What is the value of `area` now? Type `area` and the return key to see `area`'s value. If the value of `area` is not what you expected, and you can't figure out why, call your TA over for help
- Verify the current variable values by typing each variable in the shell followed by the return key. Now recalculate the `area` (using the variables) and check its value

**A function**

In the previous section, we calculated the area of a triangle. Let's define a function, called `triangle_area`, so that we can reuse the code we wrote. The general form of a function is:

```
def function_name(arguments):
    body
```

Create a new file called `triangle.py`. In this file, define a function `triangle_area`.

- arguments: `base` and `height` (Any legal variable names would do, but it's best to choose something meaningful.)
- The function should calculate the triangle's area for the given `base` and `height`, and return the area.
- Include a good docstring describing:
    - What the function does
    - What arguments it should accept, and their types
    - What it will return

Click the Run button (green triangle). In the shell, call the function as follows:

- `triangle_area(10, 12)`
- `triangle_area(3.4, 5.2)`
- `triangle_area(base, height)`

**Boolean expressions and if statements**

Write a function `storm_category(wind_speed)` in a file called `storm.py`. Your function should return the category (an `int`) that a wind speed value (an `int`) in km/h belongs to. You may assume that the wind speed value value is greater than or equal to 0.

The storm category is rated according to wind speeds greater than or equal to 0, categorized as follows:

| Wind speed | Category |
|---|---|
| < 119 km/h | 0 |
| 119-153 km/h | 1 |
| 154-177 km/h | 2 |
| 178–208 km/h | 3 |
| 209–251 km/h | 4 |
| ≥ 252 km/h | 5 |

Include a good docstring describing:

- What the function does
- What arguments it should accept, and their types
- What it will return

**PEP8 style check**

Run your `triangle.py` and `storm.py` files through the online PEP8 style checker (available on the course website under the Resources section) to see if you introduced any style errors. Correct any errors it finds.

Repeat this process until you've eliminated all style errors. Then, check your functions to make sure they still work as intended.

**Files to submit**

Submit your files `triangle.py` and `storm.py` to MarkUs under the assignment `lab2: Lab 2 week 3`